Session 10

Black Box Testing (4)

Cause Effect Graph

Black Box Testing Techniques

- Boundary Value Analysis
- Equivalence Partitioning
- Decision Table
- Cause-Effect Graph
- Combinatorial Test

Cause-Effect Graph

The Cause-Effect Graphing technique is a black-box testing technique and requirement-based technique.

Cause-Effect Graphing is very similar to Decision Table-Based Testing, where logical combinations of the inputs produce outputs; this is shown in the form of a graph.

The Cause-Effect Graphing technique begins with the set of requirements, and determines the minimum number of test cases to completely cover the requirements.

Cause-Effect Graph

Cause-effect graph models the logical relationship between program input and output which can be expressed as a Boolean expression.

Cause:any condition in the requirements that may effect the program output
(e.g. side>0,side1!=side2,month=feb)

Effect:response of a program to some combinations of input conditions
(e.g.error message displayed on the screen,
a new window is displayed,or
database is updated)

Cause-Effect Graph

Causes /effects are represented as nodes in the cause-effect graph

The graph also includes a number of intermediate nodes linking causes and effects

Procedure

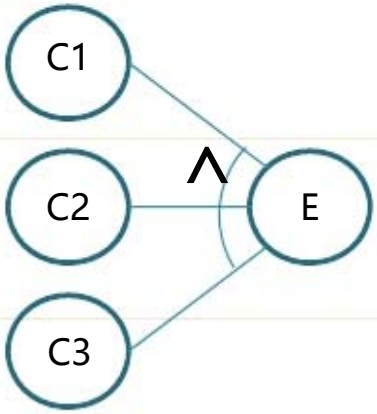
1. Identify causes and effects in specification
2. Make Boolean graph linking causes and effects.
3. Annotate impossible combinations of causes and effects (adding constraints)
4. Develop decision table from graph
5. Transform each column into test case

Notations used in cause-effect graphs

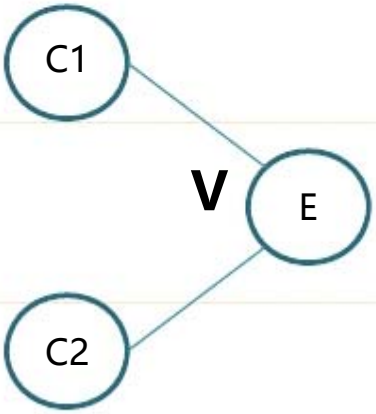
1. Identify causes and effects in specification
2. Make Boolean graph linking causes and effects

if C, then E

if ~C, then E

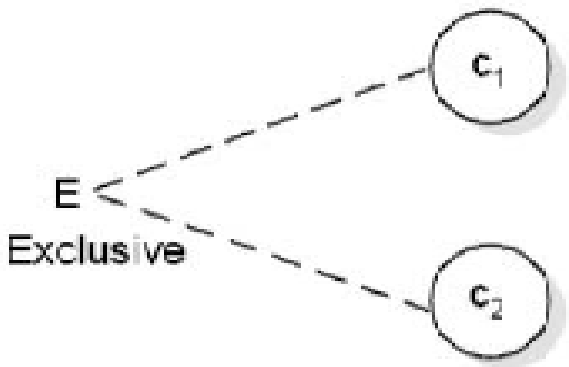


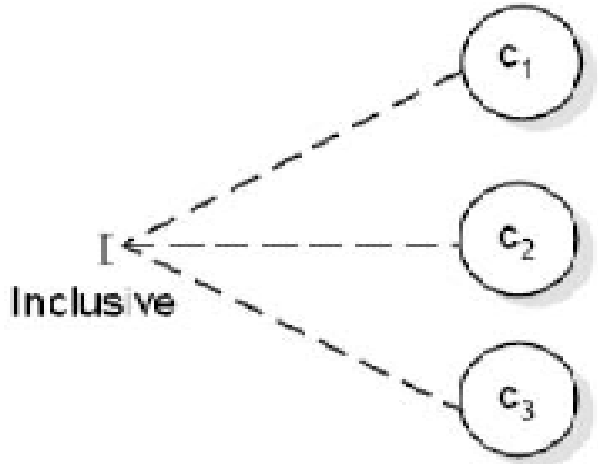
if C1 and C2 and C3, then E

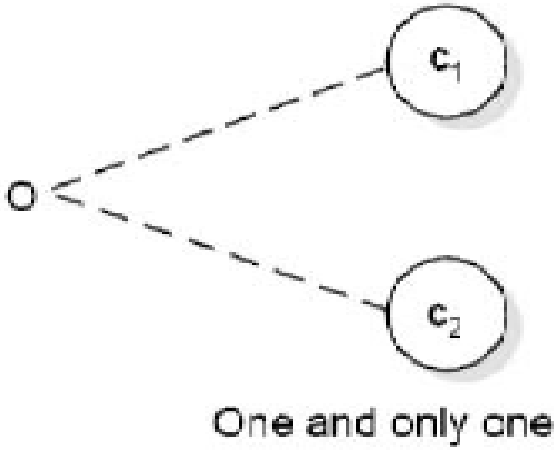


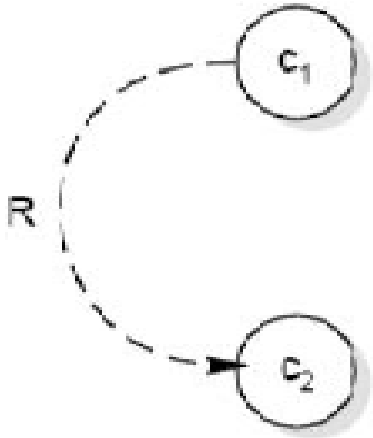
if C1 or C2, then E

3. Annotate impossible combinations of causes and effects (adding constraints)

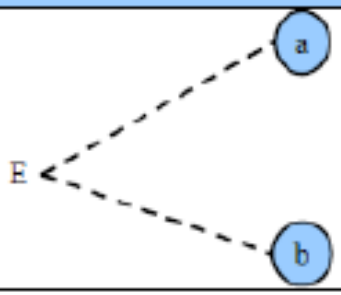
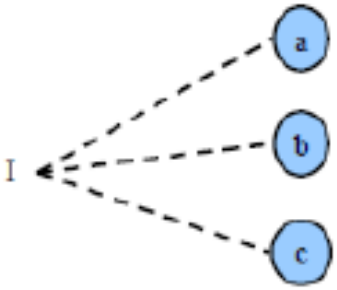
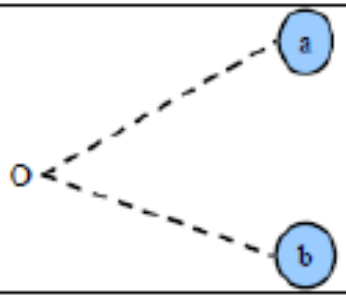
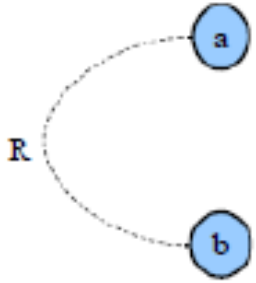
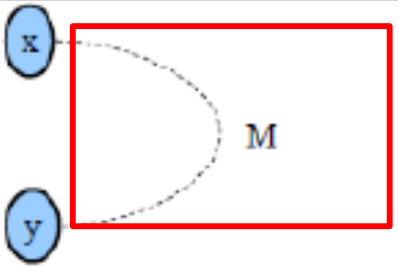








Symbol for masks constraint

Constraint Symbol	Definition
	The "E" (Exclusive) constraint states that both causes *a* and *b* cannot be true simultaneously.
	The "I" (Inclusive (at least one)) constraint states that at least one of the causes *a*, *b* and *c* must always be true (*a*, *b*, and *c* cannot be false simultaneously).
	The "O" (One and Only One) constraint states that one and only one of the causes *a* and *b* can be true.
	The "R" (Requires) constraint states that for cause *a* to be true, than cause *b* must be true. In other words, it is impossible for cause *a* to be true and cause *b* to be false.
	The "M" (mask) constraint states that if effect *x* is true; effect *y* is forced to false. (Note that the mask constraint relates to the effects and not the causes like the other constraints.

Example1

Passenger may get a discount ticket if he/she is below 12 or a student below 25.

Procedure

1. Identify causes and effects in specification
2. Make Boolean graph linking causes and effects.
3. Annotate impossible combinations of causes and effects (adding constraints)
4. Develop decision table from graph
5. Transform each column into test case

Example1

Passenger may get a discount ticket if

he/she is below 12 or a student below 25.

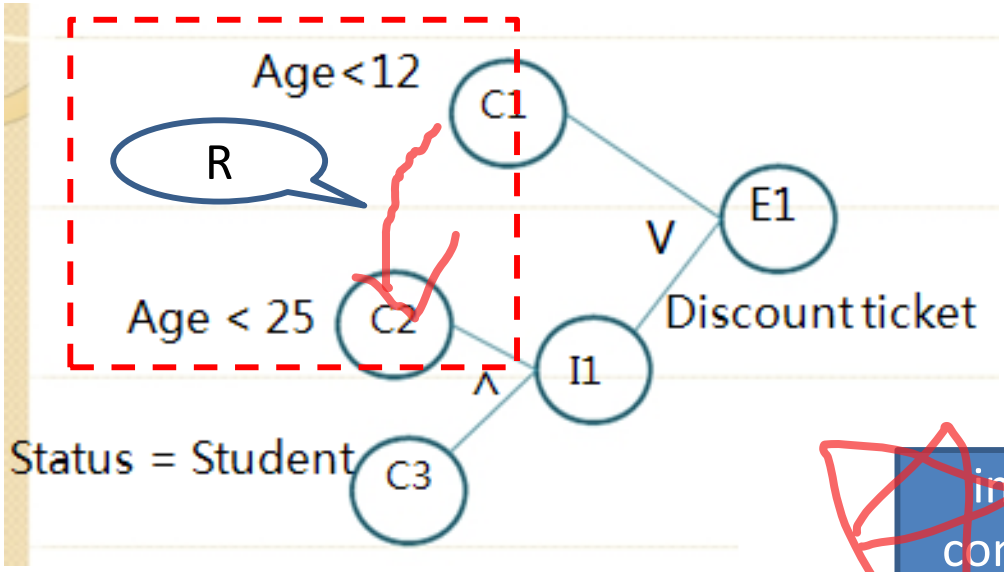
C1: Age < 12

C2: Age < 25

C3: Status = Student

I1: He/she is a student below 25

E1: Passenger can get a discount ticket



impossible combinations

	1	2	3	4	5	6	7	8
C1	T	T	T	T	F	F	F	F
C2	T	T	F	F	T	T	F	F
C3	T	F	T	F	T	F	T	F
I1	T	F	F	F	T	F	F	F
E1	T	T	T	T	T	F	F	F

Limited decision table

How about using extended decision table?

Example2

- The character in column 1 must be an ‘A’ or ‘B’. The character in column 2 must be a digit. In this situation, the file update is made. If the character in column 1 is incorrect, message x is issued. If the character in column 2 is not a digit, message y is issued.

The causes are

c_1: character in column 1 is A

c_2: character in column 1 is B

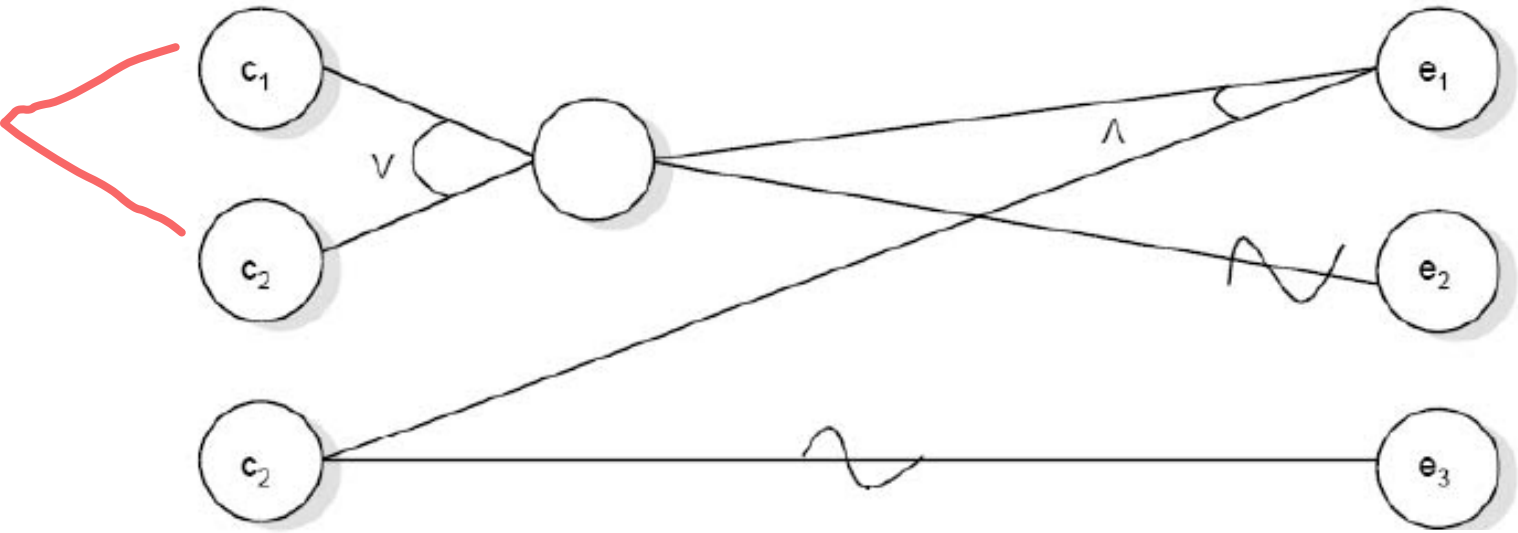
c_3: character in column 2 is a digit

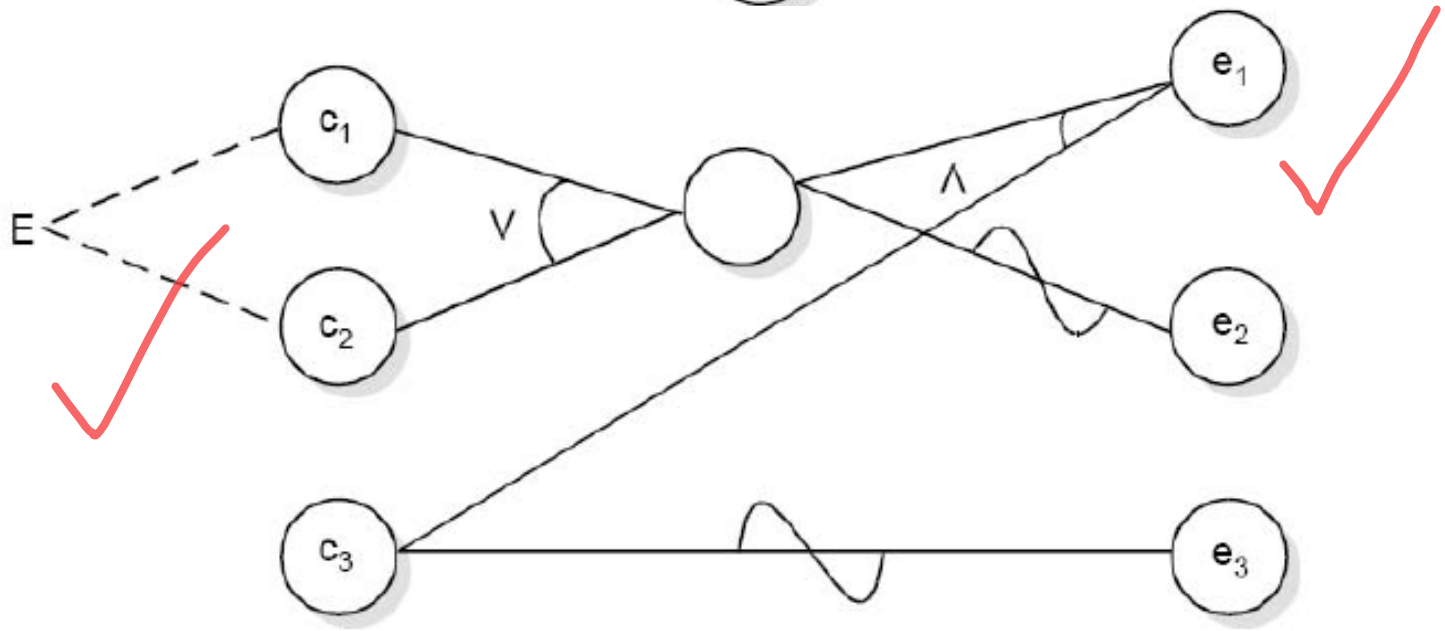
and the effects are

e_1: update made

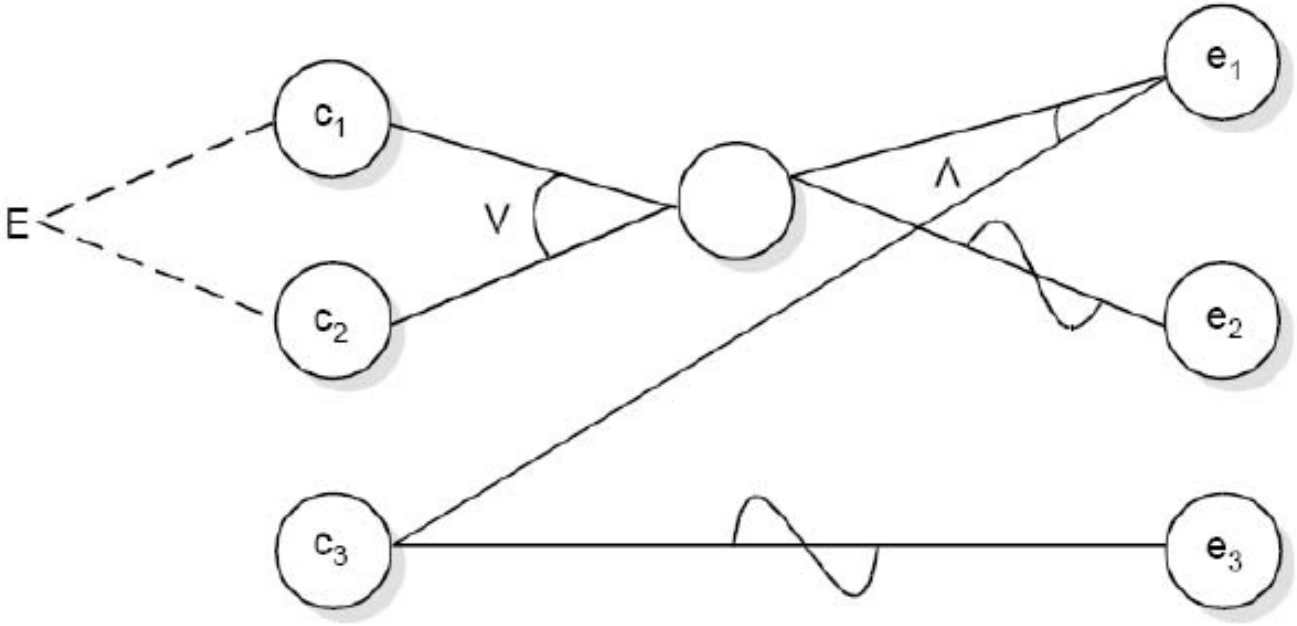
e_2: message *x* is issued

e_3: message *y* is issued





- Change cause-effect graph to decision table.



		1	2	3	4	5	6	7	8
条件（原因）	1	1	1	1	1	0	0	0	0
	2	1	1	0	0	1	1	0	0
	3	1	0	1	0	1	0	1	0
	11			1	1	1	1	0	0
动作（结果）	22			0	0	0	0	1	1
	21			1	0	1	0	0	0
	23			0	1	0	1	0	1
测试用例				A3	AM	B5	BN	C2	DY
				A8	A?	B4	B!	X6	P;

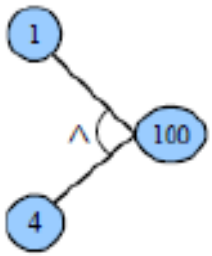
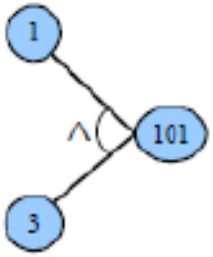
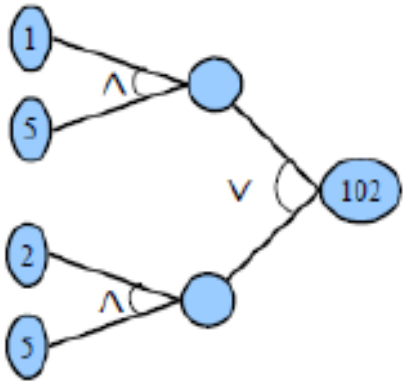
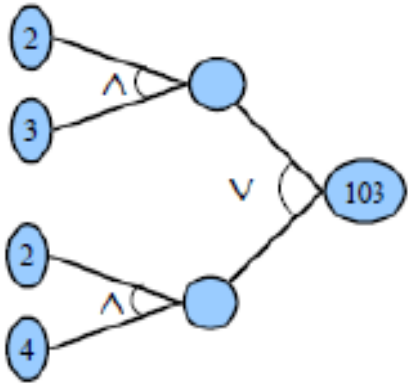
Example 3

- Requirements for Calculating Car Insurance Premiums:
  - R00101 For males between 25 and 64 years of age, the premium is $1000
  - R00102 For males less than 25 years of age, the premium is $3000
  - R00103 For anyone 65 years of age or more, the premium is $1500
  - R00104 For females less than 65 years of age, the premium is $500

- **Causes (input conditions)**
  - 1. Sex is Male
  - 2. Sex is Female
  - 3. Age is <25
  - 4. Age is >=25 and < 65
  - 5. Age is >= 65

- **Effects (output conditions)**
  - 100. Premium is $1000
  - 101. Premium is $3000
  - 102. Premium is $1500
  - 103. Premium is $500

- R00101 For males between 25 and 64 years of age, the premium is $1000
- R00102 For males less than 25 years of age, the premium is $3000
- R00103 For anyone 65 years of age or more, the premium is $1500
- R00104 For females less than 65 years of age, the premium is $500

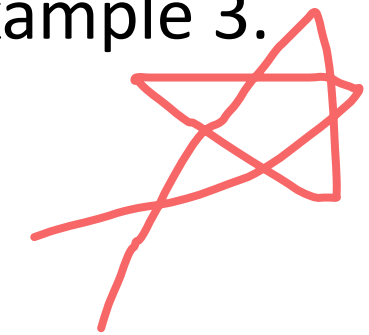
CEG	Interpretation
CEG #1: 	Causes: 1. Sex is Male and (∧) 4. Age is >=25 and < 65 Effect: 100: Premium is $1000
CEG #2: 	Causes: 1. Sex is Male and (∧) 3. Age is <25 Effect: 101: Premium is $3000
CEG #3: 	Causes: 1. Sex is Male and (∧) 5. Age is >= 65 or (∨) 2. Sex is Female and (∧) 5. Age is >= 65 Effect: 102: Premium is $1500
CEG #4: 	Causes: 2. Sex is Female and (∧) 3. Age is <25 or (∨) 2. Sex is Female and (∧) 4. Age is >=25 and < 65 Effect: 103: Premium is $500

Test Case	1	2	3	4	5	6
Causes:						
1 (male)	1	1	1	0	0	0
2 (female)	0	0	0	1	1	1
3 (<25)	1	0	0	0	1	0
4 (>=25 and < 65)	0	1	0	0	0	1
5 (>= 65)	0	0	1	1	0	0
Effects:						
100 (Premium is $1000)	0	1	0	0	0	0
101 (Premium is $3000)	1	0	0	0	0	0
102 (Premium is $1500)	0	0	1	1	0	0
103 (Premium is $500)	0	0	0	0	1	1

Test Case #	Inputs (Causes)		Expected Output (Effects)
	Sex	Age	Premium
1	Male	<25	$3000
2	Male	>=25 and < 65	$1000
3	Male	>= 65	$1500
4	Female	>= 65	$1500
5	Female	<25	$500
6	Female	>=25 and < 65	$500

Exercise-1

- Use extended decision table to test example 2.
  - The character in column 1 must be an ‘A’ or ‘B’. The character in column 2 must be a digit. In this situation, the file update is made. If the character in column 1 is incorrect, message x is issued. If the character in column 2 is not a digit, message y is issued.
  - Use extended decision table to test example 3.
  - Check out the number of rules.



Example 4

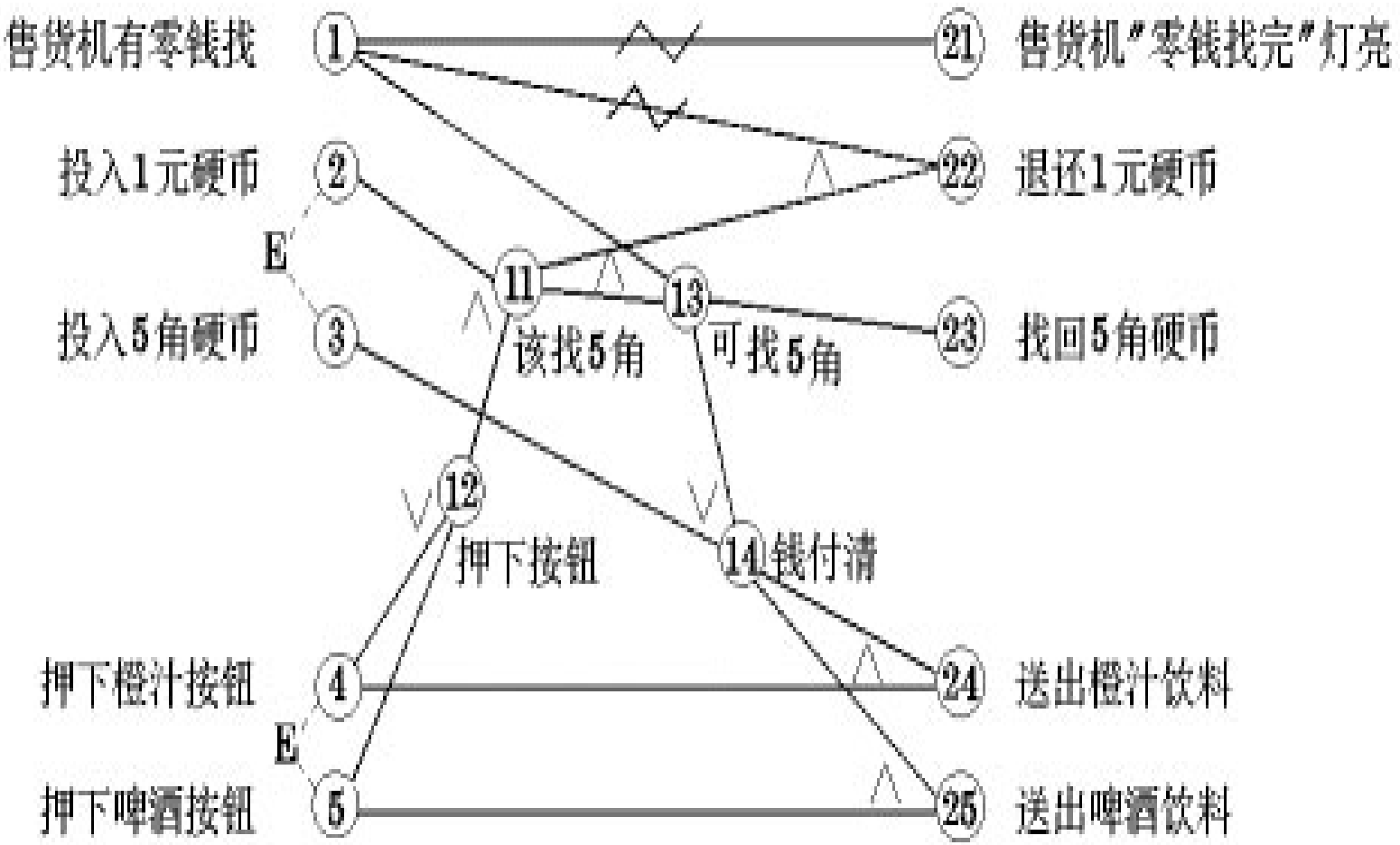
有一个处理单价为5角钱的饮料的自动售货机软件测试用例的设计。其规格说明如下：

- 若投入5角钱或1元钱的硬币，押下〖橙汁〗或〖啤酒〗的按钮，则相应的饮料就送出来。
- 若售货机没有零钱找，则一个显示〖零钱找完〗的红灯亮，这时在投入1元硬币并押下按钮后，饮料不送出来而且1元硬币也退出来；
- 若有零钱找，则显示〖零钱找完〗的红灯灭，在送出饮料的同时退还5角硬币。

- 1) 分析这一段说明，列出原因和结果
  - 原因：
  - 1.售货机有零钱找
  - 2.投入1元硬币
  - 3.投入5角硬币
  - 4.押下橙汁按钮
  - 5.押下啤酒按钮

  - 结果：
  - 21.售货机〖零钱找完〗灯亮
  - 22.退还1元硬币
  - 23.退还5角硬币
  - 24.送出橙汁饮料
  - 25.送出啤酒饮料

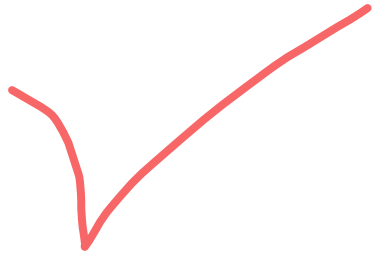
- 画出因果图，如图所示。所有原因结点列在左边，所有结果结点列在右边。建立中间结点，表示处理的中间状态。中间结点：
  - 11. 投入1元硬币且押下饮料按钮 该找5角
  - 12. 押下〖橙汁〗或〖啤酒〗的按钮 押下按钮
  - 13. 应当找5角零钱并且售货机有零钱找 可找5角
  - 14. 钱已付清 钱付清


售货机有零钱找
1
21
售货机"零钱找完"灯亮
投入1元硬币
2
22
退还1元硬币
E
11
13
投入5角硬币
3
该找5角
可找5角
23
找回5角硬币
12
押下按钮
14
钱付清
押下橙汁按钮
4
24
送出橙汁饮料
E
押下啤酒按钮
5
25
送出啤酒饮料

- 3)转换成判定表：

序号		1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1	2
条件	①	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	②	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
	③	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
	④	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
	⑤	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
中间结果	⑪						1	1	0		0	0	0		0	0	0						1	1	0		0	0	0		0	0	0
	⑫						1	1	0		1	1	0		1	1	0						1	1	0		1	1	0		1	1	0
	⑬						1	1	0		0	0	0		0	0	0						0	0	0		0	0	0		0	0	0
	⑭						1	1	0		1	1	1		0	0	0						0	0	0		1	1	1		0	0	0
结果	㉑						0	0	0		0	0	0		0	0	0						1	1	1		1	1	1		1	1	1
	㉒						0	0	0		0	0	0		0	0	0						1	1	0		0	0	0		0	0	0
	㉓						1	1	0		0	0	0		0	0	0						0	0	0		0	0	0		0	0	0
	㉔						1	0	0		1	0	0		0	0	0						0	0	0		1	0	0		0	0	0
	㉕						0	1	0		0	1	0		0	0	0						0	0	0		0	1	0		0	0	0
测试用例							Y	Y	Y		Y	Y	Y		Y	Y							Y	Y	Y		Y	Y	Y		Y	Y	

- 4) 在判定表中，阴影部分表示因违反约束条件的不可能出现的情况，删去。
- 第16列与第32列因什么动作也没做，也删去。最后可根据剩下的16列作为确定测试用例的依据。

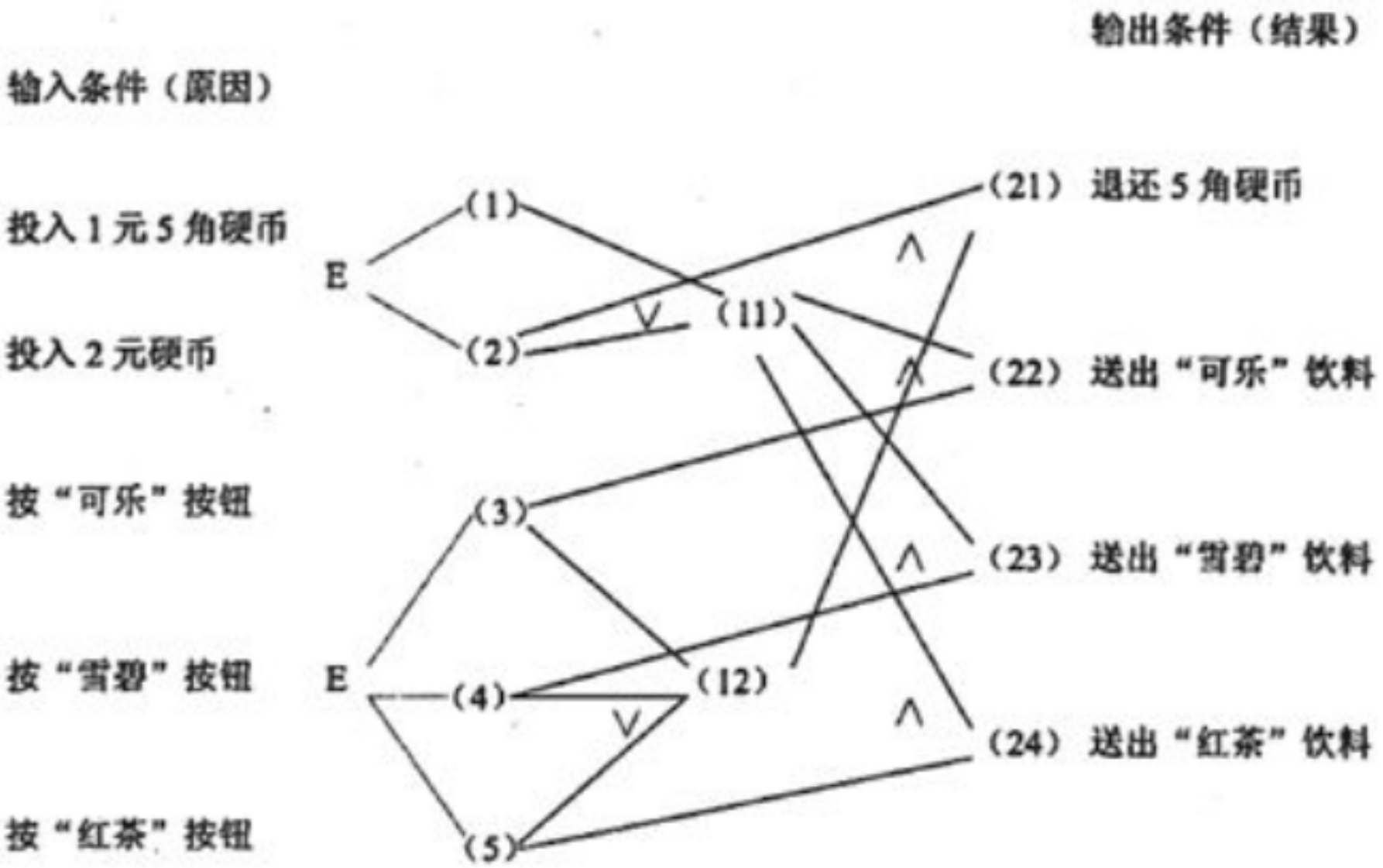


Exercise-2

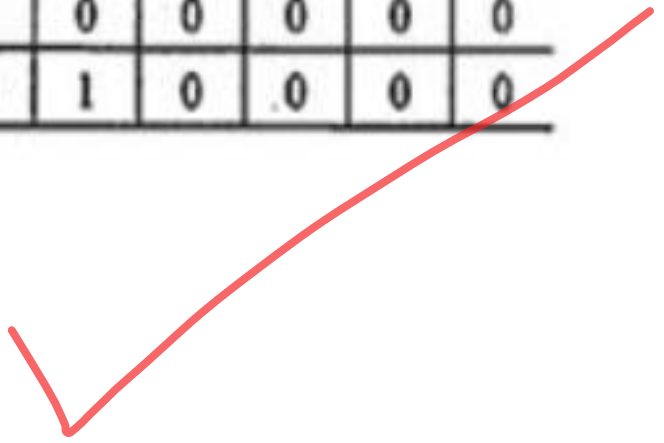
- Other version of vending machine.
  - 有一个处理单价为1元5角钱的盒装饮料的自动售货机。若投入1元5角硬币，按下“可乐”、“雪碧”或“红茶”按钮，相应的饮料就送出来。若投入的是2元硬币，在送出饮料的同时退还5角硬币。

- 因:
  - 1、投入1元5角硬币
  - 2、投入2元硬币
  - 3、按下‘可乐’按钮
  - 4、按下‘雪碧’按钮
  - 5、按下‘红茶’按钮

- 果:
  - 1、退出5角硬币
  - 2、送出‘可乐’
  - 3、送出‘雪碧’
  - 4、送出‘红茶’


输出条件（结果）
输入条件（原因）
投入1元5角硬币
投入2元硬币
按“可乐”按钮
按“雪碧”按钮
按“红茶”按钮
E
E
(1)
(2)
(3)
(4)
(5)
(11)
(12)
(21) 退还5角硬币
(22) 送出“可乐”饮料
(23) 送出“雪碧”饮料
(24) 送出“红茶”饮料

			1	2	3	4	5	6	7	8	9	10	11
输入	投入1元5角硬币	(1)	1	1	1	1	0	0	0	0	0	0	0
	投入2元硬币	(2)	0	0	0	0	1	1	1	1	0	0	0
	按“可乐”按钮	(3)	1	0	0	0	1	0	0	0	1	0	0
	按“雪碧”按钮	(4)	0	1	0	0	0	1	0	0	0	1	0
	按“红茶”按钮	(5)	0	0	1	0	0	0	1	0	0	0	1
中间结点	已投币	(11)	1	1	1	1	1	1	1	1	0	0	0
	已按钮	(12)	1	1	1	0	1	1	1	0	1	1	1
输出	退还5角硬币	(21)	0	0	0	0	1	1	1	0	0	0	0
	送出“可乐”饮料	(22)	1	0	0	0	1	0	0	0	0	0	0
	送出“雪碧”饮料	(23)	0	1	0	0	0	1	0	0	0	0	0
	送出“红茶”饮料	(24)	0	0	1	0	0	0	1	0	0	0	0



Exercise-3

- Given the requirements as follows:
  - For international airlines of Europe and America, all classes have in-flight meals and in-flight entertainments;
  - For other international airlines, all classes have in-flight meals, only business classes have in-flight entertainments;
  - For domestic airlines, business classes have in-flight meals, but no in-flight entertainments;
  - For domestic airlines, economy classes have in-flight meals if the flying time is longer than 2 hours, but no in-flight entertainments.
- Design the corresponding **extended** **entry decision table:**

Black Box Testing Techniques

- Boundary Value Analysis
- Equivalence Partitioning
- Decision Table
- Cause-Effect Graph
- Combinatorial Test